NFC in Linux

How to get started with the PN7120S controller board

Public

MobileKnowledge
July 2015

Agenda

Session 8th July: *PN7120 - Best plug'n play full NFC solution*

- PN7120 & NXP NFC product families
- PN7120 use cases and target markets
- PN7120 comparison to NFC frontends and previous NFC controller solutions
- PN7120 product details
- PN7120 product support package

Session 15th July: *NFC in Linux - How to get started with the PN7120S controller board*

- NFC tools for Linux
- NXP solutions for Linux
- How to get started with Raspberry-Pi and PN710S
- How to integrate Linux libnfc-nci SW stack into a Linux system

What is Linux

- An operating system
- Free and open-source software development and distribution
  - Great community support
- Usually packaged in the form of Linux distributions
  - They include
    - The Linux kernel
    - Other software components (libraries, tools…), depending on the intended use of the distribution
  - Examples: Debian, Ubuntu, Red Hat…
- Due to its license policy and its great flexibility -> ported to more computer hardware platforms than any other operating system
  - Used in PCs, servers, smart devices, embedded systems…
- Example of operating system based on the Linux kernel: Android

NFC in Linux

Some NFC tools (1)

Linux libnfc-nci

- Developed by NXP
- Derived from the available and proven Android stack
- The recommended stack for the **PN7120** NFC controller

Linux NFC

- Development led by Intel
- Aims:
  - Complete
  - Open source
  - Hardware independent

Open NFC

- Mainly developed by Inside secure
- Designed to be portable to different operating systems

NFC in Linux

Some NFC tools (2)

libnfc

- Academic project
- Open source and community support
- Supports various operating systems

nfcpy

- Sponsored by Sony
- Python module for NFC
- Considered as the NFC Forum reference implementation

M.U.S.C.L.E.
Movement For The Use Of Smart Cards In A Linux Environment

PCSC-Lite

- Developed inside the M.U.S.C.L.E. project
- Open source implementation of PC/SC
- Aim: interact with smart cards

NFC in Linux

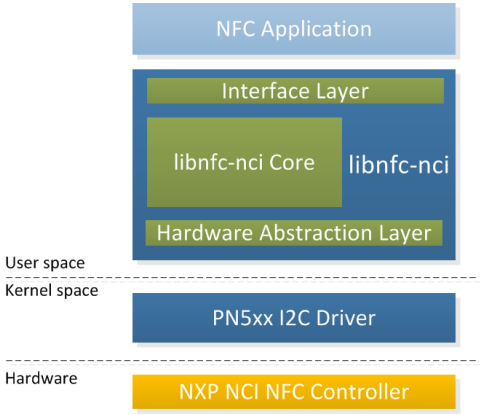
Some NFC tools - Features

	Host Interfaces	Tag R/W	LLCP	Handover	Card Emulation
Linux libnfc-nci	NCI	Yes	NPP, SNEP, Handover	Bluetooth, WiFi	Yes
Linux NFC	HCI, NCI, USB	Yes	NPP, SNEP, Handover, PHDC	Bluetooth, WiFi	Yes
Open NFC	HCI	Yes	SNEP, Handover	Bluetooth, WiFi	Yes
libnfc	USB, UART	Yes	No	No	Yes
nfcpy	USB	Yes	SNEP, Handover, PHDC	Bluetooth, WiFi	No

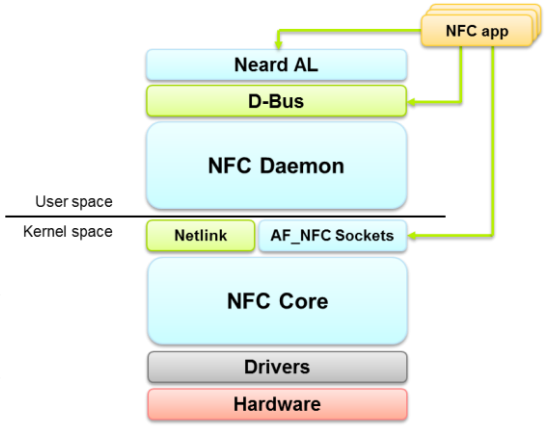
PN7120 in Linux

- PN7120 is the brand-new full NFC Forum-compliant controller from NXP
  - Supports CE, R/W and P2P modes
  - Compatible with ISO/IEC 14443-A&B, FeliCa and ISO/IEC 15693 card
  - Integrated firmware with NCI interface
- PN7120 is supported by the Linux libnfc-nci and Linux NFC stacks
  - Linux NFC
    - Maintained by the community
    - NXP does not provide support
  - Linux libnfc-nci
    - The recommended stack for the PN7120
- Documentation about the integration of the PN7120 and the Linux libnfc-nci stack into a Linux environment is available at the NXP website:

  http://www.nxp.com/products/identification_and_security/nfc_and_reader_ics/nfc_controller_solutions/PN7120A0EV.html#documentation



Linux libnfc-nci stack



Linux NFC stack

Documentation	
UM10819	PN7120 User Manual
AN11697	PN7120 Linux SW stack integration guidelines
Linux NFC API Guide	Linux libnfc-nci stack APIs description for the PN7120.

The Linux libnfc-nci stack

- The NXP NFC stack for Linux systems
- Derived from the available and proven Android stack
  - Very robust and complete due to its maturity
- Supports the implementation of a broad range of use cases in a Linux environment
  - NFC Forum tag types, P2P, handover, HCE, raw commands…
- Works together with the PN5xx I2C driver from NXP, which offers communication with NXP NFC controllers through an I2C interface
- It is the recommended stack for the brand-new **PN7120** NFC controller
- Both the Linux libnfc-nci stack and the PN5xx I2C driver are distributed by GitHub:
  - Linux libnfc-nci stack: https://github.com/NXPNFCLinux/linux_libnfc-nci
  - PN5xx I2C driver: https://github.com/NXPNFCLinux/nxp-pn5xx

The Linux libnfc-nci stack

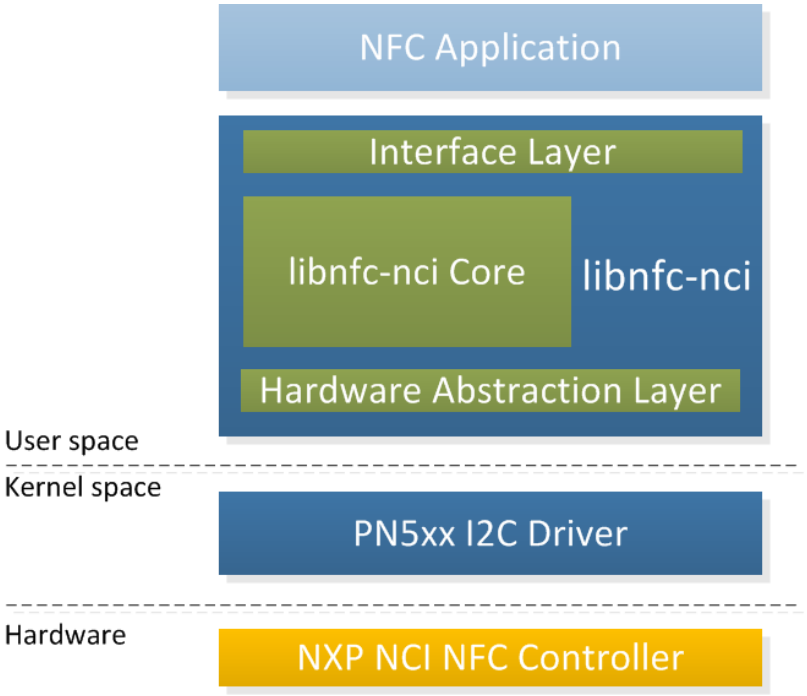
Features

- NDEF tag support
- MIFARE Classic tag support
- P2P, LLCP, SNEP
- WiFi & BT handover
- Raw tag command support
- Proprietary NCI command support
- Host Card Emulation support

*It is planned that NXP PN7120 successors will also be supported by this stack

The Linux libnfc-nci stack

The architecture



Interface Layer: exposes the library API to the user application.

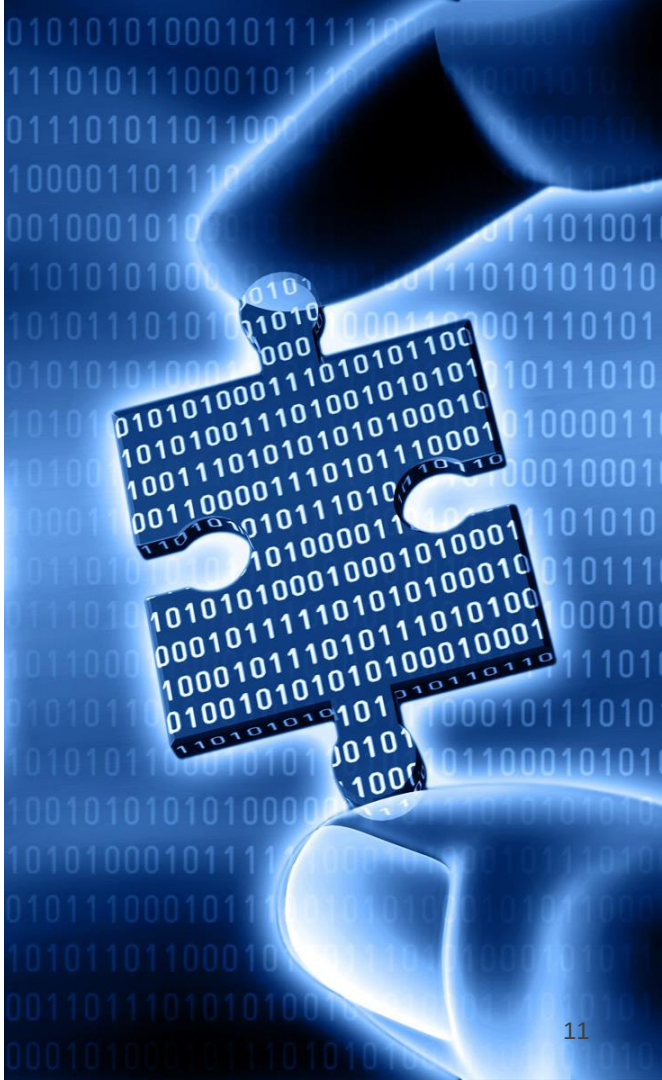
Libnfc-nci Core: implements the NFC functionality (NCI, NDEF, LLCP and SNEP protocols, tag operations, Host Card Emulation…)

Hardware Abstraction Layer: provides connection to the kernel driver as well as basic functionalities like self-test or firmware update

PN5xx I2C Driver: offers communication to the NFC controller connected over the I2C physical interface

The Linux libnfc-nci stack
The API

- **C** API exposed by the Interface Layer of the Linux libnfc-nci stack
- Based on **callback** functions that are executed when an event occurs
  - Different possible events, e.g., tag arrival, tag departure, handover request received, command received for an emulated card…
  - In read/write mode, the callback returns a **handle** to the tag that allows the application to interact with it (in other modes it is not necessary)
- Functions and types declared in the documents:
  - **linux_nfc_api.h**: the main document. It exposes the NFC features that will be used by the end application
  - **linux_nfc_factory_api.h**: dedicated to end device production. It can be used to check the antenna connection, for CE/FCC certification…
- More information in the ***Linux_NFC_API_Guide.html*** document inside the *doc* sub-folder of the stack delivery



The Linux libnfc-nci stack

The API – linux_nfc_api.h

Some functions

- extern void nfcManager_registerTagCallback(nfcTagCallback_t *callback);
  /* register a tag callback functions.*/

- extern void nfcManager_enableDiscovery (int technologies_masks, int reader_only_mode, int enable_host_routing, int restart);
  /* start nfc discovery.*/

- extern int ndef_readText(unsigned char *ndef_buff, unsigned int ndef_buff_length, char * out_text, unsigned int out_text_length);
  /* read text message from NDEF data. */

- extern int nfcTag_readNdef(unsigned int handle, unsigned char *ndef_buffer, unsigned int ndef_buffer_length, nfc_friendly_type_t *friendly_ndef_type);;
  /* read NDEF message from tag. */

- extern int nfcTag_transceive (unsigned int handle, unsigned char *tx_buffer, int tx_buffer_length, unsigned char* rx_buffer, int rx_buffer_length, unsigned int timeout);
  /* send raw command to tag. */

Some types

- typedef struct {
  unsigned int technology; /* the technology of tag */
  unsigned int handle; /* the handle of tag */
  …
  } nfc_tag_info_t; /* NFC tag information structure definition. */

- typedef struct {
  int is_ndef; /* the flag to indicate if it contains NDEF record */
  unsigned int current_ndef_length; /* existing NDEF message length */
  …
  } ndef_info_t; /* NFC NDEF message information structure definition. */

- typedef struct {
  void (*onTagArrival) (nfc_tag_info_t *pTagInfo); /* NFC tag callback function when tag is detected. */
  void (*onTagDeparture) (void) /* NFC tag callback function when tag is removed. */
  } nfcTagCallback_t; /* NFC tag callback function structure definition. */

The Linux libnfc-nci stack

The API – writing tag

1. Initialize the Linux libnfc-nci library
2. Register for tag discovery
3. Start NFC discovery

Tap the tag

4. Tag discovery is notified
5. Write NDEF message to the Tag

This example depicts the steps to follow from SW perspective to be able to write NDEF content (e.g. web address or business card) to an NFC Tag.

The Linux libnfc-nci stack

The API –WiFi pairing

Start broadcasting WiFi

1. Initialize the Linux libnfc-nci library
2. Register for device discovery
3. Start NFC discovery

Tap the NFC-enabled smartphone

4. Phone discovery is notified

5. Send WiFi credentials to the phone

6*. Pop-up is raised on the phone proposing to connect to the WiFi network

Phone connects to the WiFi network

This example depicts the steps to follow from SW perspective to be able to send WiFi credentials to an NFC phone.

* Starting from Android 5.0 (LolliPop) WIFI configuration NDEF record, as defined by WiFi alliance, is natively supported.

The Linux libnfc-nci stack

The API – BT handover

Start BT OOB server

1. Initialize the Linux libnfc-nci library
2. Register for device discovery
3. Start NFC discovery

Select a picture, tap the smartphone and beam it

4. Device discovery is notified

5*. Handover Request message is received from the phone

6. Send Handover Select record to the phone

Phone connects via BT and transmit the picture to the OOB server

This example depicts the steps to follow from SW perspective in order to receive a picture over Bluetooth from an NFC phone.

* Starting from Android 4.0 (IceCreamSandwich) Bluetooth handover, as defined by NFC Forum, is natively supported for large files exchange.

The Linux libnfc-nci stack

The API – an example application

```
#include <stdio.h>
#include <string.h>
#include <semaphore.h>
#include <linux_nfc_api.h>

static sem_t sem;
static int mTagHandle;
```

```
/* Callback for the tag arrival event */
void onTagArrival (nfc_tag_info_t *pTag)
{
    mTagHandle = pTag->handle;
    sem_post(&sem);
}
```

```
/* Callback for the tag departure event */
void onTagDeparture (void ){}
```

```
/* Main function: waits for an NFC tag and shows if it contains an NDEF
message*/
int main(int argc, char *argv[])
{
    printf("\n########## Our first Linux libnfc-nci application
##########\n\n");

    /* Initialize variables */
    sem_init(&sem, 0, 0);
    nfcTagCallback_t tagCb;
    tagCb.onTagArrival = onTagArrival;
    tagCb.onTagDepature = onTagDeparture;

    /* Initialize stack */
    nfcManager_doInitialize ();
    nfcManager_registerTagCallback(&tagCb);
    nfcManager_enableDiscovery(DEFAULT_NFA_TECH_MASK, 1, 0, 0);

    /* Wait for tag */
    printf("Waiting for an NFC tag...\n\n");
    sem_wait(&sem);
```

```
    /* Check if the tag contains an NDEF message */
    printf("Tag found!\n");
    ndef_info_t info;
    memset(&info, 0, sizeof(ndef_info_t));
    nfcTag_isNdef(mTagHandle, &info);
    if (info.is_ndef)
        printf("The tag contains an NDEF message\n\n");
    else
        printf("The tag does not contain an NDEF message\n\n");

    /* Deinitialize stack */
    sem_destroy(&sem);
    nfcManager_disableDiscovery();
    nfcManager_deregisterTagCallback();
    nfcManager_doDeinitialize ();

    return 0;
}
```

The Linux libnfc-nci stack

The configuration files

- Allow developers to configure several parameters of the stack
- Read during the initialization of the stack
- Two files:
  - **libnfc-brcm.conf**: allows the user to configure:
    - Log levels
    - Host listening enabled
    - Polling technologies
    - P2P listening technologies
  - **libnfc-nxp.conf**: allows the user to configure:
    - Log levels
    - MIFARE Classic reading enabled
    - System clock
    - Polling profile
    - Other NXP NFC controller settings
- Both files are self-explanatory

Example – polling technologies

```
################################################################################
# Force tag polling for the following technology(s).
# The bits are defined as tNFA_TECHNOLOGY_MASK in nfa_api.h.
# Default is NFA_TECHNOLOGY_MASK_A | NFA_TECHNOLOGY_MASK_B |
#           NFA_TECHNOLOGY_MASK_F | NFA_TECHNOLOGY_MASK_ISO15693 |
#           NFA_TECHNOLOGY_MASK_B_PRIME | NFA_TECHNOLOGY_MASK_KOVIO |
#           NFA_TECHNOLOGY_MASK_A_ACTIVE | NFA_TECHNOLOGY_MASK_F_ACTIVE.
#
# Notable bits:
# NFA_TECHNOLOGY_MASK_A           0x01    /* NFC Technology A           */
# NFA_TECHNOLOGY_MASK_B           0x02    /* NFC Technology B           */
# NFA_TECHNOLOGY_MASK_F           0x04    /* NFC Technology F           */
# NFA_TECHNOLOGY_MASK_ISO15693    0x08    /* Proprietary Technology     */
# NFA_TECHNOLOGY_MASK_KOVIO       0x20    /* Proprietary Technology     */
# NFA_TECHNOLOGY_MASK_A_ACTIVE    0x40    /* NFC Technology A active mode */
# NFA_TECHNOLOGY_MASK_F_ACTIVE    0x80    /* NFC Technology F active mode */
# This flag when set to zero will disable Reader mode.
POLLING_TECH_MASK=0xEF
```

NXP solutions for Linux

NXP NFC readers supported in Linux

Product	Description	Tools supported	More info
PN532	NFC controller with integrated firmware. It supports the SPI, I²C and HSU host interfaces.	Linux NFC, libnfc, nfcpy	http://www.nxp.com/products/identification_and_security/nfc_and_reader_ics/nfc_controller_solutions/PN5321A3HN.html
PN533	USB NFC controller with integrated firmware.	Linux NFC, libnfc, nfcpy	http://www.nxp.com/products/identification_and_security/nfc_and_reader_ics/nfc_controller_solutions/PN5331B3HN.html
PN7120	Full NFC Forum-compliant NFC controller with integrated firmware and NCI interface.	Linux libnfc-nci, Linux NFC	http://www.nxp.com/products/identification_and_security/nfc_and_reader_ics/nfc_controller_solutions/PN7120A0EV.html
PR533	USB NFC controller with integrated firmware. It supports the CCID protocol over the USB link.	PCSC-Lite	http://www.nxp.com/products/identification_and_security/nfc_and_reader_ics/nfc_controller_solutions/PR5331C3HN.html
PN512	Full NFC Forum-compliant NFC frontend.	NXP NFC Reader Library	http://www.nxp.com/products/identification_and_security/nfc_and_reader_ics/nfc_frontend_solutions/series/PN512.html

PN7120 controller SBC Kit

OM5577/PN7120S

- Demoboard for the PN7120 NFC controller
- Designed to work with Raspberry-Pi or BeagleBone
  - Can be adapted to other systems
- Drivers available for Linux and Android

- Linux images available for Raspberry-Pi and BeagleBone
  - They come with the full Linux libnfc-nci stack installed
  - They integrate the PN7120 drivers
  - They include demo software

* Board available at NXP distributors

Demoboard website PN7120S/OM5577: www.nxp.com/demoboard/OM5577.html

Package ID	Package details
OT334610	Hardware design files for OM5577/PN7120S
Software	
OM5577_Rbi	Demonstration image for quick start in Raspberry Pi Linux environment
OM5577_BBB_Linux	Demonstration image for quick start in BeagleBone Linux environment
OM5577_BBB_Kitkat	Demonstration image for quick start in BeagleBone Android environment
Documentation	
UM10878	PN7120 NFC Controller SBC Kit user manual
AN11646	PN7120 NFC Controller SBC Kit quick start guide
PN7120_SBC_Kit	PN7120 NFC Controller SBC Kit OM5577/PN7120S (leaflet)

Quick start guides

Getting started with the PN7120 controller board

What you need

Raspberry-Pi

PN7120
controller board

Raspberry-Pi
interface board

SD card
(at least 4GB)

Power supply

HDMI screen

USB mouse
and keyboard

Internet
connection

Getting started with the PN7120 controller board

Hands-on video

http://youtu.be/e1-KhQPxNU4

Integrating the Linux libnfc-nci stack into a Linux system

- Integration described in the application note AN11697 – PN7120 Linux Software Stack Integration Guidelines
- It consists of 2 main steps:
  - Installing the **PN5xx I2C driver** as part of the kernel
  - Installing the **Linux libnfc-nci stack** in user mode
- Steps to install the driver:
  - 1.- Download the Linux kernel source code
  - 2.- Download the driver source code
  - 3.- Include the driver in the kernel compilation
  - 4.- Indicate to the kernel where and how to access the new hardware
  - 5.- Build the kernel
  - 6.- Install the new kernel
- Steps to install the library:
  - 1.- Download the library source code
  - 2.- Build the library
  - 3.- Install the library

Integrating the Linux libnfc-nci stack into a Linux system

Hands-on video

http://youtu.be/TCgCRi-tKxM

Integrating the Linux libnfc-nci stack into a Linux system

Summary – Integrating the stack into a Raspbian system

Installing the driver

1.- Download the Linux kernel source code

```
uname –r
wget https://github.com/raspberrypi/linux/archive/rpi-3.18.y.tar.gz
tar xvfz rpi-3.18.y.tar.gz
```

2.- Download the driver source code

```
cd linux-rpi-3.18.y/drivers/misc
git clone https://github.com/NXPNFCLinux/nxp-pn5xx.git
```

3.- Include the driver in the kernel compilation

vi Makefile **(add:** obj-y +=nxp-pn5xx/ **)**

vi Kconfig **(add:** source "drivers/misc/nxp-pn5xx/Kconfig“ **)**

4.- Indicate to the kernel where and how to access the new hardware

```
cd ~/linux-rpi-3.18.y/arch/arm/boot/dts
cat /proc/cpuinfo
vi bcm2708-rpi-b.dts ( add: /include/ “bcm270x-pn7120.dtsi” )
cp ~/linux-rpi-3.18.y/drivers/misc/nxp-pn5xx/sample_devicetree.txt ./bcm270x-pn7120.dtsi
vi bcm270x-pn7120.dtsi ( modify according to your platform )
```

5.- Build the kernel

```
cd ~/linux-rpi-3.18.y
make bcmrpi_defconfig
make menuconfig ( include the driver )
make zImage modules dtbs
```

6.- Install the new kernel

```
sudo make modules_install
sudo cp arch/arm/boot/dts/*.dtb /boot/
sudo cp arch/arm/boot/dts/overlays/*.dtb /boot/overlays/
sudo scripts/mkknlimg arch/arm/boot/zImage /boot/kernel.img
sudo reboot
cd /etc/udev/rules.d
sudo gedit pn5xx_i2c.rules ( add: ACTION=="add", KERNEL=="pn544", MODE="0666" )
```

Installing the library

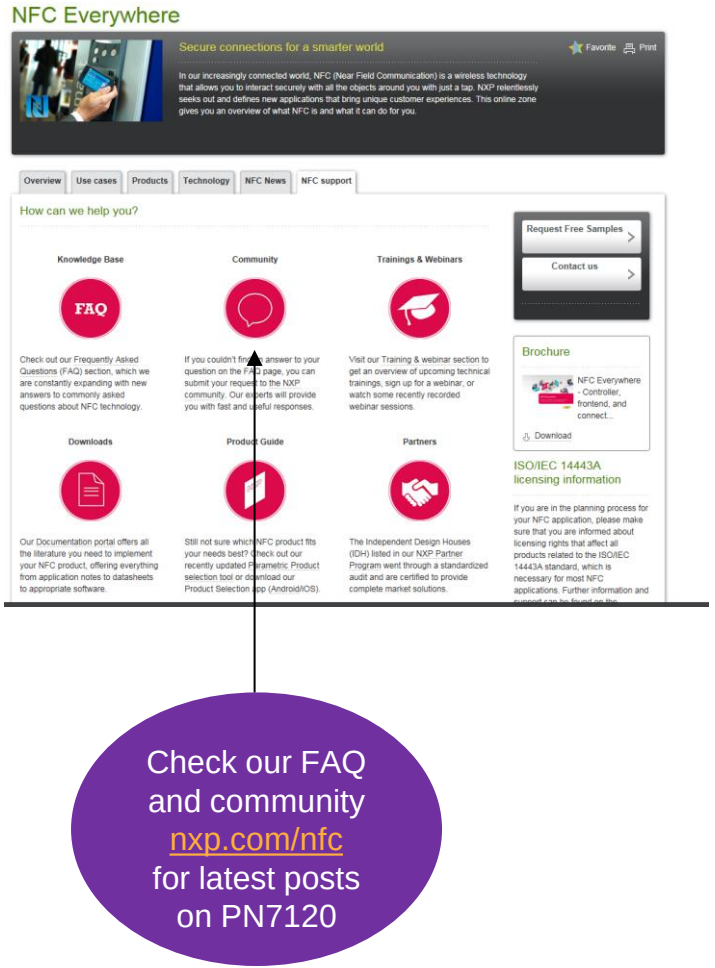
```
git clone https://github.com/NXPNFCLinux/linux_libnfc-nci.git
cd linux_libnfc-nci
./bootstrap
./configure –sysconfdir=/etc
make
sudo make install
```

Wrap up

- There are several tools to work with NFC devices in Linux environments
- The **recommended stack** to develop software for the **PN7120** NFC controller is the Linux libnfc-nci stack from NXP
- The **Linux libnfc-nci** stack:
  - Is derived from the available and proven Android stack
  - Supports the implementation of a broad range of use cases in a Linux environment (NFC tags, P2P, handover , HCE…)
  - Works over the **PN5xx I2C driver** from NXP
  - Provides an API based on callback functions
- NXP supports the development of NFC applications in Linux environments through:
  - Drivers for its NFC controllers
  - Demoboards
- The **PN7120 NFC controller board** is a great tool for getting started with the Linux libnfc-nci stack and with the PN7120 NFC controller

Further information

- NFC Everywhere
  http://www.nxp.com/nfc
- NFC Everywhere support page
  http://www.nxp.com/techzones/nfc-zone/community.html
- Here you can check out the community for FAQs or post your question into the discussion forum for NFC Readers
- PN7120 product support information
  http://www.nxp.com/products/identification_and_security/nfc_and_reader_ics/nfc_controller_solutions/PN7120A0EV.html
- PN7120 controller board support information
  http://www.nxp.com/demoboard/OM5577.html
- NXP Linux libnfc-nci stack
  https://github.com/NXPNFCLinux/linux_libnfc-nci
- NXP PN5xx I2C driver
  https://github.com/NXPNFCLinux/nxp-pn5xx

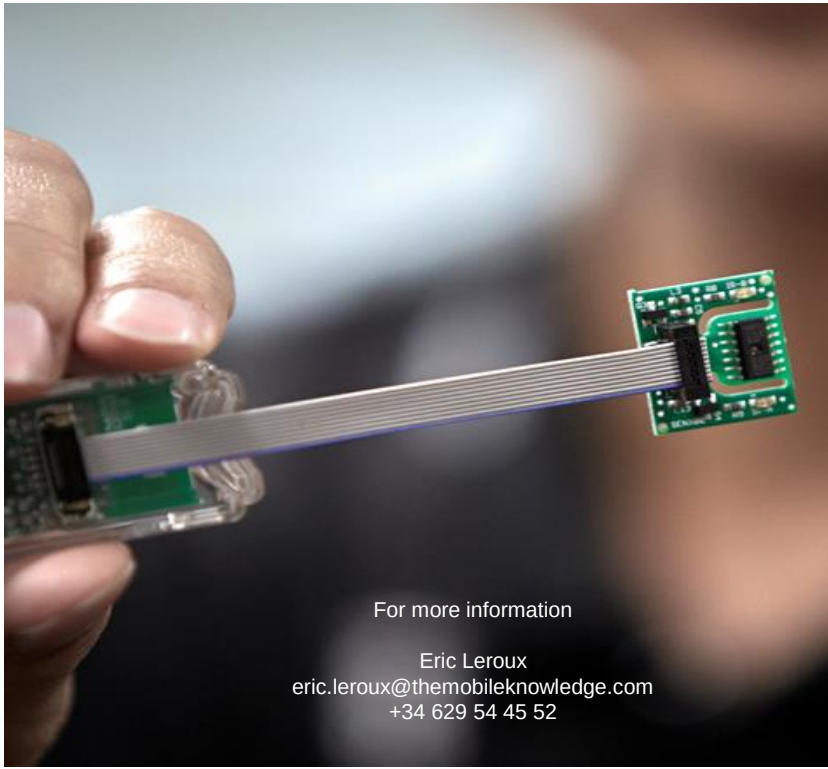


MobileKnowledge

Thank you for your attention

MobileKnowledge
www.themobileknowledge.com

- We are a global competence team of hardware and software technical experts in all areas related to contactless technologies and applications.
- Our services include:
  - Application and system Design Engineering support
  - Project Management
  - Technological Consulting
  - Advanced Technical Training services
- We address all the exploding identification technologies that include NFC, secure micro-controllers for smart cards and mobile applications, reader ICs, smart tags and labels, MIFARE family and authentication devices.



NXP Training Mobile Knowledge

NFC in Linux

Franz Van-Horenbeke (Speaker) / Eric Leroux (Host)

Thank you for your kind attention!

- Please remember to fill out our evaluation survey (pop-up)
- Check your email for material download and on-demand video addresses
- Please check NXP and MobileKnowledge websites for upcoming webinars and training sessions

  www.nxp.com/products/related/customer-training.html

  www.themobileknowledge.com/content/knowledge-catalog-0

Training Mobile Knowledge

Thank you